Linux Network Servers

TCP/IP Parte 2

Objetivo: Conhecer mais de TCP/IP, mais sobre os arquivos de configuração e configuração de subredes. Aprender ARP e monitoração de tráfego com iptraf e usar o tcpdump.

Antes de vermos novos comandos, vamos começar estudando vários arquivos de configuração do sistema.

Todo computador tem um nome, chamado de hostname. Para configuramos o nome de nosso hostname, temos que editar o arquivo /etc/hostname.

```
ebl:~# cat /etc/hostname
ebl
```

Lembra do comando **ping? Vamos pingar nossa própria máquina:**

```
ping nome-da-sua-maquina
ebl:~# ping -c4 ebl
PING ebl.debian (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.052 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.050 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.054 ms
64 bytes from localhost (127.0.0.1): icmp_seq=4 ttl=64 time=0.047 ms

--- ebl.debian ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.047/0.050/0.054/0.009 ms
```

O -c indica o número de repetições.

Curiosidade:

Para desativar a resposta ao ping faça:

```
# echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Repare que o nome da sua máquina foi convertido para 127.0.0.1. Essa conversão é feita pelo arquivo /etc/hosts.

```
ebl:~# cat /etc/hosts
127.0.0.1       localhost
127.0.0.1       ebl.debian      ebl
```

Edite o /etc/hosts e acrescente a seguinte linha no final do arquivo:

```
127.0.0.1       teste
```

Linux Network Servers

E depois pingue o nome teste:

ebl:~# ping -c4 teste

Existem agora vários nomes que convertem para o IP 127.0.0.1.

Nós podemos trocar o nosso hostname. Para trocar, basta abrir o arquivo com o vim, editar e salvar. O nome não vai ser trocado imediatamente, para isso faça o seguinte:

ebl:~# echo "seu-novo-hostname" > /proc/sys/kernel/hostname

Abra um segundo terminal e você verá que o hostname foi trocado.

Será que é possível identificar um sistema operação somente pelo ping?

O campo TTL mostrado na saída do comando ping significa Time To Live, que significa o número de saltos entre máquinas que os pacotes podem demorar numa rede de computadores antes de serem descartados (máx. 255). Diferentes sistemas operacionais usam diferentes valores para o TTL. Se o TTL apresentado por 128, provavelmente é uma máquina Windows. Se for 64, provavelmente é Linux.

Mas qual a função do TTL?

O TTL evita que os pacotes ao serem enviados fiquem em loop infinito, isto é, evita que a banda seja consumida de forma não necessária.

Roteamento é um processo que visa encaminhar pacotes de uma rede para outra. Quando o pacote deve passar de uma sub-rede para outra, o responsável por esse repasse é o roteador. O roteador pode ser via software ou hardware. O repasse de pacotes ocorre segundo o modelo hop-by-hop (salto-por-salto).

O roteador recebe o pacote e verifica qual é o destino através do cabeçalho IP e calcula o próximo salto. Este processo se repete até que o pacote seja entregue ao destinatário. Com o comando traceroute no Linux é possível verificar o número de roteadores envolvidos em uma comunicação.

Linux Network Servers

Mas como o número de saltos pode ser observado?

Exemplo:
$ traceroute www.google.com

traceroute to www.google.com (209.85.193.104), 30 hops max, 40 byte packets

1 192.168.0.1 (192.168.0.1) 0.345 ms 0.371 ms 0.406 ms
2 200.175.172.1 (200.175.172.1) 10.752 ms 10.793 ms 10.834 ms
3 mxin1.gvt.com.br (200.175.157.129) 11.419 ms 11.461 ms 13.863 ms
4 gvt-ge-0-1-2-rc02.bsa.gvt.net.br (189.59.250.17) 24.231 ms 26.680 ms gvt-ge-0-1-2-rc01.bsa.gvt.net.br (189.59.250.13) 27.115 ms
5 gvt-ge-4-0-4-rc02.spo.gvt.net.br (189.59.248.5) 41.202 ms gvt-ge-4-0-0-rc01.bsa.gvt.net.br (189.59.250.1) 28.616 ms gvt-ge-4-0-4-rc02.spo.gvt.net.br (189.59.248.5) 43.866 ms
6 gvt-ge-0-0-0-rc01.spo.gvt.net.br (189.59.248.1) 44.324 ms 28.804 ms gvt-ge-4-0-4-rc02.spo.gvt.net.br (189.59.248.5) 29.769 ms
7 gvt-ge-0-0-0-rc01.spo.gvt.net.br (189.59.248.1) 30.393 ms 31.102 ms gvt-ge-1-0-0-205-rc01.spo.gvt.net.br (189.59.240.2) 32.797 ms
8 gvt-ge-1-0-0-205-rc01.spo.gvt.net.br (189.59.240.2) 34.273 ms 209.85.249.232 (209.85.249.232) 35.518 ms 36.470 ms
9 209.85.251.214 (209.85.251.214) 42.619 ms 209.85.249.232 (209.85.249.232) 39.155 ms 40.374 ms
10 209.85.251.210 (209.85.251.210) 49.273 ms 49.706 ms *

$ ping -c1 www.google.com
PING www.l.google.com (209.85.193.104) 56(84) bytes of data.
64 bytes from br-in-f104.google.com (209.85.193.104): icmp_seq=1 ttl=246 time=30.3 ms

Executei o ping para pegar o valor de ttl (time to live) do destinatário: 246.
Para cada roteador que um pacote percorrer, é decrementado em "1" o valor do campo TTL do destinatário.

Cada sistema operacional tem ser valor de TTL, segue uma tabela padrão:

Windows XP = 255
Linux = 64
Linux com firewall = 255

Observação: O administrador do sistema pode modificar o TTL para dificultar ataques de fingerprint (determinar o sistema operacional da máquina-alvo para estruturar ataque cracker).

255 - X = 246
X = 9
Logo, o pacote percorreu 9 roteadores.

Linux Network Servers

Verificar o valor de TTL é um dos passos para determinar o sistema operacional (fingerprint) em um pen-test (teste de penetração) na máquina-alvo. Mas só a análise do TTL não garante a identidade do SO, por isso outras técnicas são utilizadas.

No Linux, o ttl é setado no arquivo:
/proc/sys/net/ipv4/ip_default_ttl

Se você for alterar o ttl, não coloque um valor muito pequeno como, por exemplo, 10, isso irá atrapalhar o acesso a redes muito distantes geograficamente. Mantenha o padrão!

Pessoal, vimos como funciona o /etc/hosts. Mas e se eu precisar converter um nome que não está nele, como isso é feito?

O /etc/resolv.conf é o arquivo de configuração principal do código do resolvedor de nomes. Seu formato é um arquivo texto simples com um parâmetro por linha e o endereço de servidores DNS externos são especificados nele.

Como exemplo, o /etc/resolv.conf se parece com isto:

```
nameserver 192.168.12.1
```

O nameserver é o IP do servidor de DNS que irá resolver os nomes na rede para nós.

Na aula passado usamos o ifconfig para ver nossas interfaces de rede. Agora onde no sistema essas configurações ficam salvas?

No Debian/Ubuntu ficam em /etc/network/interfaces

Vamos ver a cara desse arquivo, essa é uma configuração para um IP estático:

```
auto eth0
iface eth0 inet static
        address 192.168.2.10
        netmask 255.255.255.0
   broadcast 192.168.2.255
        gateway 192.168.2.1
```

Essa para um IP dinâmico:

```
auto eth0
iface eth0 inet dhcp
```

Veja como está sua configuração:

```
ebl:~# cat /etc/network/interfaces
```

Linux Network Servers

Dica:

No Red Hat o arquivo de configuração de rede fica em /etc/sysconfig/network-scripts/.

Bom, agora que já conhecemos melhor nosso sistema, vamos dar uma olhada na rede com o comando **arp**.

Lembra que nossa placa de rede tem um identificador chamado endereço MAC? Veja qual é o MAC da sua placa:

```
ebl:~# ifconfig eth0 | grep HW
eth0      Link encap:Ethernet  HWaddr 00:15:c5:32:e0:4b
```

Bem, quando todos estamos ligados em um switch, cada placa tem seu MAC. Para uma máquina conversar com a outra, o sistema tem que associar um MAC a um IP, pra saber onde mandar os pacotes. Esse é protocolo ARP. Address Resolution Protocol ou ARP é um protocolo usado para encontrar um endereço Ethernet (MAC) a partir do endereço IP. O ARP atua na camada 3 do modelo OSI. O emissor difunde em broadcast um pacote ARP contendo o endereço IP de outro host e espera uma resposta com um endereço MAC respectivo. Devido aos pacotes broadcast terem um alto custo em termos de banda de rede, cada host guarda um cache com os endereços conhecidos.

Para sabermos os MACs que estão em cache:

```
ebl:~# arp -a
? (192.168.1.103) at 00:13:72:02:0b:a8 [ether] on eth0
? (192.168.1.1) at 00:13:10:d5:48:f8 [ether] on eth0
```

O RARP (reverse ARP) faz o contrário do ARP. O objetivo do RARP é comunicar a partir do endereço MAC. O RARP é usando quando um host precisa obter sua configuração via DHCP.

O comando arping é um software que utiliza o ARP ao invés do ping para descobrir se outras máquinas da rede estão ativas.

Mesmo máquinas protegidas por firewall ou configuradas para não responder a ping respondem a pacotes ARP. Isso é muito útil na hora de detectar problemas na rede.

```
# aptitude install arping
```

O arping só funciona na rede local, pois quando ele é usando em um endereço externo (internet), ao passar no roteador, os campos com endereço ARP são removidos.

Faça um teste!

```
# arping www.uol.com.br
```

Dica de segurança!

Use o arpwatch para monitorar sua rede. Ele irá te alertar você caso alguém ligue um novo computador em sua rede, ou troque o endereço MAC. Vamos imaginar a seguinte situação: Um funcionário espertinho leva um laptop e conecta ele na rede, afim de levar arquivos da empresa embora. O arpwatch estará monitorando a rede.

Linux Network Servers

Porém, seria possível clonar o endereço MAC de uma máquina?
É possível sim! E é muito fácil!

```
ifconfig eth0 down hw ether 00:00:00:00:00:01
ifconfig eth0 up
```

Temos associado um IP a um MAC, podemos ter mais de IP para o mesmo MAC?
Sim, usando aliases!

```
ebl:~# ifconfig eth0:0 192.168.1.105 netmask 255.255.255.0
```

Para acrescentarmos um IP a uma interface, usamos a sintaxe eth0:X, onde X é numero que representa o alias.

Agora vou mostrar um exemplo prático de criação de subredes.

Qual é a vantagem de segmentar uma rede maior em outras menores???

Segmentar uma rede em sub-redes é interessante quando é necessário otimizar o tráfego de rede e possuir um controle maior sobre ela.

Pode-se dividir a rede de acordo com os setores de um empresa por exemplo.

Exemplo:
Considere o seguinte endereço de rede 192.168.1.0/24.
24 é a máscara de rede = 255.255.255.0
24 é no modo octal
Essa rede será dividida em dois segmentos.
Para isso, é necessário trabalhar apenas com o último octeto da máscara de rede.

O último octeto da máscara da classe C é 0.
0 em binário é 00000000 (oito bits).
Para aumentar o número de sub-redes é necessário acrescentar o bit 1 no último octeto.
Para dividir uma rede em dois segmentos precisa-se fazer que a máscara de rede seja: 11111111.11111111.11111111.10000000 (25 bits setados com 1)

Regra:
Número de redes = 2 elevado ao número de bits em 1 = 2 elevado 1 = 2
Número de hosts = 2 elevado ao número de bits em 0 - 2 = 2 elevado 7 - 2 = 128 - 2 = 126

Subtrai-se 2 de 128, pois um endereço é para rede o outro é para o broadcast.
Logo, a regra é: 2 elevado a N = X

No qual X é o número de subredes que quero.

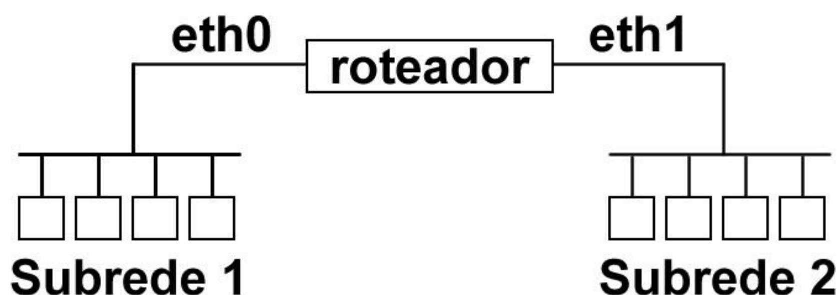
Se quero duas subredes:
2 elevado a N = 2
Logo, N=1

Linux Network Servers

Então, precisa-se de 1 bit setado no último octeto.

Subrede 1
Endereço de rede: 192.168.1.0
Broadcast: 192.168.1.127
Hosts variando de 1 a 126
Máscara: 192.168.1.0/25

Subrede 2
Endereço de rede: 192.168.1.128
Broadcast: 192.168.1.255
Hosts variando de 129 a 254
Máscara: 192.168.1.128/25



Vamos agora configurar o roteamento estático!

Configuração das duas interfaces do roteador:

ifconfig eth0 192.168.1.1 netmask 255.255.255.128 broadcast 192.168.1.127
ifconfig eth1 192.168.1.129 netmask 255.255.255.128 broadcast 192.168.1.255

Caso se tenha apenas uma interface, podemos usar o ip virtual e configurar dois endereços para essa interface, conforme exemplo abaixo:

ifconfig eth0 192.168.1.1 netmask 255.255.255.128 broadcast 192.168.1.127
ifconfig eth0:0 192.168.1.129 netmask 255.255.255.128 broadcast 192.168.1.255

Lembrando que o comando ifconfig só mantem as configurações na placa enquanto o sistema estiver ligado, se você reiniciar a máquina as configurações são perdidas.

No Debian, você pode armazenar as configurações como ip, máscara de rede etc em /etc/network/interfaces.

Consulte se seu sistema permite repasse entre as interfaces de rede. Para isso, o valor da diretriz ip_forward deve ser 1:
sysctl -a | grep ip_forward

Execute o comando ping para o endereço de broadcast:
ping 192.168.1.255

Podemos alterar em tempo de execução caso seja necessário:
sysctl -w net.ipv4.ip_forward=1
ou
echo 1 > /proc/sys/net/ipv4/ip_forward

Visualize se alterou:
cat /proc/sys/net/ipv4/ip_forward

Linux Network Servers

Será que é possível fazer que isso fique de forma permamente?

Habilitando o Forward (repasse) de pacotes de forma que fique permanente!

Para isso basta inserir uma entrada no arquivo /etc/sysctl.conf:

echo net.ipv4.ip_forward=1 >> /etc/sysctl.conf

sysctl -p /etc/sysctl.conf

cat /proc/sys/net/ipv4/ip_forward

Temos agora que configurar as estações nas sub-redes!

Configurando um host na subrede 192.168.1.0/25:
ifconfig eth0 192.168.1.2 netmask 255.255.255.128 broadcast 192.168.1.127

Para a rede 192.168.1.0/25, o Gateway deve ser 192.168.1.1:
route add default gw 192.168.1.1
route -n
netstat -atun

Configurando um host na subrede 192.168.1.128/25:
ifconfig eth0 192.168.1.130 netmask 255.255.255.128 broadcast 192.168.1.255

Para a rede 192.168.1.128/25, o Gateway deve ser 192.168.1.129:
route add default gw 192.168.1.129
route -n
netstat -atun

Linux Network Servers

Agora que já aprendemos mais sobre rede, vamos aprender mais comandos.

Quando precisamos monitorar a rede, utilizamos o iptraf.

Instale-o caso não tenha:
ebl:~# aptitude install iptraf

Execute o comando:
iptraf

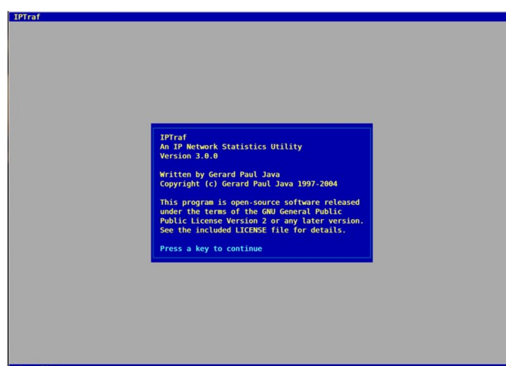


Ilustração 1: Execução iptraf

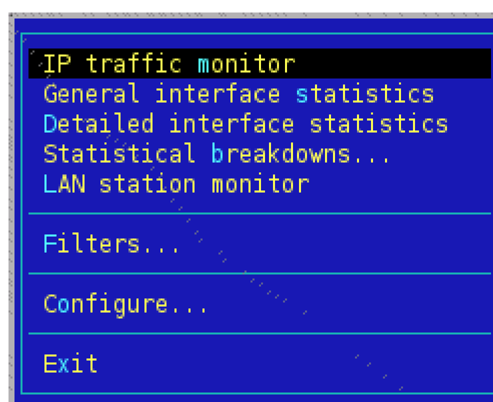


Ilustração 2: Monitorar o tráfego

Linux Network Servers

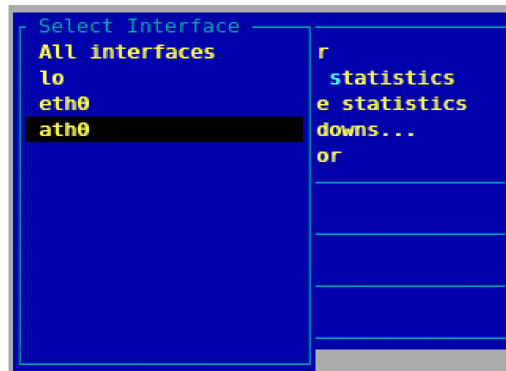


Ilustração 3: Escolhendo a interface

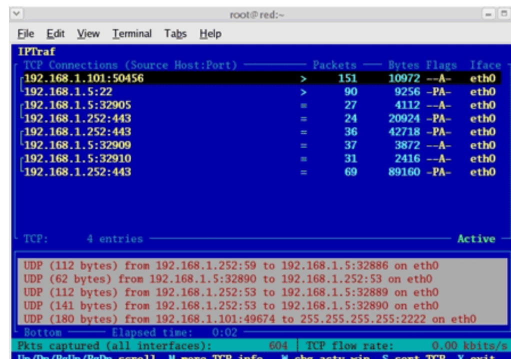


Ilustração 4: iptraf monitorando a rede

Nós vimos no iptraf as conexões que estão ativas, temos como ver mais a fundo e observar o que está sendo enviado/recebido?

É perfeiramente possível, usando o tcpdump.

tcpdump -X -i eth0

O tcpdump vai jogar tela todo o tráfico que estiver passando por sua interface de rede. Repare que não existe criptrografia!

Conexões de origem podem ser monitoradas utilizando o parâmetro src host, um exemplo simples seria monitorarmos o tráfego que vem de 192.168.1.5 para nosso computador, com o ip 192.168.1.3. A linha de comando ficaria da seguinte forma:

tcpdump -i eth0 src host 192.168.1.5

Linux Network Servers

Se quisermos monitorar as conexões especificando um host de destino, poderíamos fazê-lo com o parâmetro dst host, o exemplo abaixo mostra todo o tráfego do host 192.168.1.3 com 192.168.1.1, no caso, 192.168.1.1 é nosso gateway.

```
# tcpdump -i eth0 dst host 192.168.1.1
```

No tcpdump podemos também especificar portas de origem e destino com os comandos src port e dst port, um exemplo seria monitorarmos o tráfego destinado à porta 80 (http), para isso utilizaríamos a linha de comandos abaixo e navegaríamos em um site qualquer:

```
# tcpdump -i eth0 dst port 80
```

Se em uma rede local é possível usar o tcpdump, seria possível utilizá-lo em uma rede Wi-Fi?

Perfeitamente! Por padrão o tráfego de uma rede sem fio é totalmente desprotegido!

A primeira linha de defesa de uma rede Wi-Fi é a adoção de criptografia. Com ela, os dados que trafegam entre o computador e o roteador wireless são codificados.

A criptografia WPA (Wireless Protected Access) e a mais recente WPA2 já são mais usadas do que a tecnologia WEP (Wireless Encryptation Protocol). Por isso, opte pela WPA ou WPA2 se possível – o protocolo WEP é relativamente fácil de ser quebrado.

Mas lembre-se: utilize o mesmo protocolo em todos os dispositivos da rede; não é possível misturá-los. As chaves utilizadas pelo WPA e WPA2 são mudadas dinamicamente, o que as torna praticamente impossível de serem quebradas. Apesar disso, utilize uma senha difícil de ser descoberta como chave de criptografia, em uma combinação de letras e números com pelo menos 14 caracteres. Se o roteador utilizado é de um modelo mais antigo que suporte apenas o protocolo WEP, a alternativa é utiliza uma chave WEP de 128 bits ou trocar o hardware.